A Siamese CNN for Image Steganalysis

Weike You, Hong Zhang, *Member, IEEE*, and Xianfeng Zhao, *Senior Member, IEEE*

***Abstract*— Image steganalysis is a technique for detecting data hidden in images. Recent research has shown the powerful capabilities of using convolutional neural networks (CNN) for image steganalysis. However, due to the particularity of steganographic signals, there are still few reliable CNN-based methods for applying steganalysis to images of arbitrary size. In this paper, we address this issue by exploring the possibility of exploiting a network for steganalyzing images of varying sizes without retraining its parameters. On the assumption that natural image noise is similar between different image sub-regions, we propose an end-to-end, deep learning, novel solution for distinguishing steganography images from normal images that provides satisfying performance. The proposed network first takes the image as the input, then identifies the relationships between the noise of different image sub-regions, and, finally, outputs the resulting classification based upon them. Our algorithm adopts a Siamese, CNN-based architecture, which consists of two symmetrical subnets with shared parameters, and contains three phases: preprocessing, feature extraction, and fusion/classification. To validate the network, we generated datasets composed of steganography images with multiple sizes and their corresponding normal images sourced from BOSSbase 1.01 and ALASKA #2. Experimental results produced by the data generated by various methods show that our proposed network is well-generalized and robust.**

***Index Terms*— Steganalysis, deep learning, Siamese convolutional neural network.**

I. Introduction

THERE are currently trillions of images available on the internet. People use these images to document their lives, share their feelings and satisfy other interests. Unfortunately, due to image steganography software development, criminal groups can efficiently deliver messages through regular image transfers. Because these software programs attempt to make the payload look like random image noise produced by the camera sensor and circuitry, detection of imagery associated with criminal activities using the human eye alone is impossible. The danger to public security posed by this technology is a real-world issue. Therefore, researchers have investigated and developed innovative image steganalysis techniques for detecting and countering this threat.

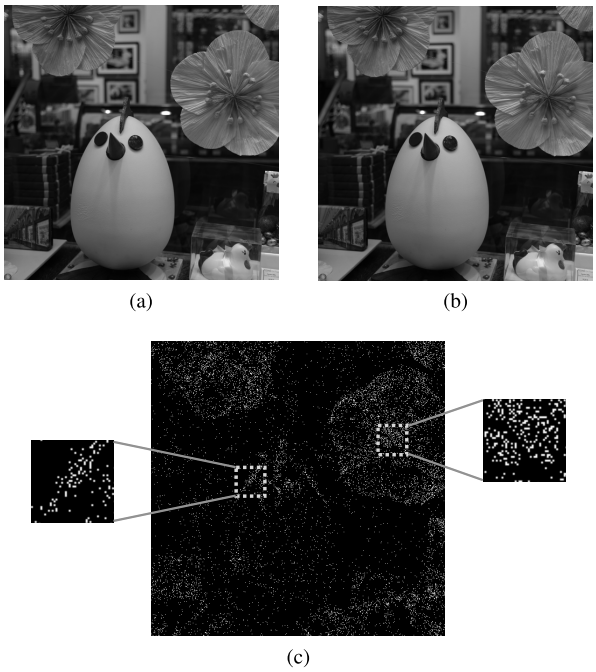


Fig. 1. A cover image and the corresponding stego image. Fig. (a) presents the original image, while Fig. (b) is the steganographic image generated by Holub and Fridrich [2]. The pixel-by-pixel differences between Fig. (a) and Fig. (b) are shown in Fig. (c); these are hard to detect using the human eye.

Image steganography - hiding data within an image - is one of the most common techniques in covert communication [1]. Successful image steganography algorithms (interested readers should consult [2]–[6]) usually embed secret messages into the complex areas of images to avoid causing visible artifacts and changes in statistical properties. One such example is shown in Fig. 1: Fig. 1(a) is the original image, and Fig. 1(b) is the steganographic image generated by a typical steganography method [2]. The pixel-by-pixel differences between Fig. 1(a) and Fig. 1(b) are shown in Fig. 1(c). However, Fig. 1(a) and Fig. 1(b) will appear identical to the human eye. (To simplify the following discussion, we name the original image and corresponding steganographic image "cover" and "stego", respectively.)

Previous classical image steganalysis approaches have focused on the statistical properties of hand-crafted features between cover and stego images [7]–[15]. Researchers have adopted various description techniques for combining information attained by evidence estimators. For example, Fridrich and Kodovský [16] show that both a medium-dimensional

Manuscript received November 17, 2019; revised March 29, 2020, May 30, 2020, and July 3, 2020; accepted July 6, 2020. Date of publication July 30, 2020; date of current version August 10, 2020. This work was supported in part by the National Natural Science Foundation of China under Grant 61972390, Grant 61802393, and Grant U1736214, in part by the National Key Research and Development Program of China under Grant 2019QY0701, and in part by the Climbing Program of the Institute of Information Engineering of the Chinese Academy of Sciences. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Patrick Bas. *(Corresponding author: Hong Zhang.)*

The authors are with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China, and also with the School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100093, China (e-mail: youweike@iie.ac.cn; zhanghong@iie.ac.cn; zhaoxianfeng@iie.ac.cn).

Digital Object Identifier 10.1109/TIFS.2020.3013204

feature set with a Gaussian SVM and a high-dimensional feature set with an Ensemble Classifier can lead to a significant improvement in detection accuracy for all tested algorithms. However, since every description technique has its limitations [17], methods based on traditional machine learning are limited by their manually-defined features.

Over the past five years [18], deep learning has emerged as a promising framework providing state-of-the-art performance for image steganalysis [17], [19]–[28]. For instance, Boroumand *et al.* [28] show that a steganographic signal can be extracted by CNN-based architecture even without fixed, high-pass filters. However, because resizing the image decreases the signal-to-noise ratio between the cover image and the steganographic signal [29], to the best of our knowledge there are currently only a few state-of-the-art approaches that can steganalyze arbitrary-size images. In a method developed by Tsang and Fridrich [29], a fixed number of statistical moments of the feature maps is extracted before the fully-connected part. Experimental results have demonstrated the effectiveness of this approach. It is a practical and ingenious solution; however, with large, realistic, and highly heterogeneous datasets there is a slight decrease in accuracy in its performance (this will be illustrated in Section IV-C). Yousfi *et al.* [30] and Butora and Fridrich [31] implement the same strategy.

In this paper, we propose an end-to-end, highly discriminative neural network for differentiating steganographic images from original images, one which provides a more satisfying performance in the evaluation of images of various sizes. Our network adopts a Siamese, CNN-based architecture, which is used to capture relationships between image sub-regions by using two supervisory signals simultaneously; those are then used as the basis for classification. This shall be explained in more detail in Section III. We can exploit these clues on the assumption that steganographically embedded changes in different sub-areas - themselves generated by satisfying steganography methods - are not the same. These internal differences can provide some content-independent information for our network. For the purposes of training and evaluating our proposed model, as well as generating a training dataset, we use four image steganography methods: WOW [5], S-UNIWARD [2], HILL [3], and the non-adaptive Least-Significant-Bit Replacement (LSBR) [32]. The cover images were sourced from two databases: BOSSbase 1.01 [33] and ALASKA #2 [34], [35]. In order to facilitate future image steganalysis research, details of both the code and the generated dataset will be released [1].

The main contributions of this study are summarized as follows:

- A novel steganalysis method for the image of arbitrary size is proposed, which is solely based on a Siamese convolutional neural network (CNN) that exploits the relationships between image sub-regions.
- Our proposed Siamese, deep-learning framework is a unique solution for distinguishing stego images from cover images. The performance of this network is comparable with that of state-of-the-art image steganalysis approaches for steganalyzing fixed-size images.
- The methodology of our Siamese framework could guide further research into making existing networks handle heterogeneous datasets more effectively.

The rest of the paper is organized as follows: in Section II, we briefly review related work addressing several relevant respects; in Section III, we present our Siamese CNN-based model, and discuss its underlying objectives; in Section IV, we present extensive experimental results; and, in Section V, we draw conclusions on the findings presented in this paper.

II. Related Work

Our aim is to present a novel, effective solution for steganalyzing images of arbitrary size. Accordingly, in this section we review relevant, typical image steganography approaches and existing CNN-based steganalysis methods.

A. Common Image Steganography Approaches

The intention underlying image steganography approaches is to effect an invisible change in order to conceal a secret message in a cover image. Even if a third party discovers the stego, suspicions concerning the data hidden inside it are unlikely because it has the appearance of a normal image. The most common image steganography approaches can be broadly categorized into three classes: naive steganography [32], [36], adaptive steganography [2]–[6], and deep learning-based embedding [37]–[44].

Naive steganography methods are relatively simple and are used widely across the Internet for entertainment purposes. These are also the methods that create the most easily discernible artifacts. For instance, the Least-Significant-Bit (LSB) technique [32], [36] embeds a secret message into the cover image by altering the values of pixels without measuring the distortion. As a result, they can be easily attacked simply by using the prior statistical knowledge of cover images [1].

Adaptive steganography is currently the most practical method. It not only improves security by embedding secret messages into more textured areas of the cover images [2]–[6], but it also uses efficient steganographic codes, *e.g.* Syndrome Trellis Codes (STCs) [45], in order to minimize the total impact of the embedded changes. For instance, Pevný *et al.* [4] defined a weighted difference of feature vectors used in steganalysis in their analysis of distortion in their Highly Undetectable steGO (HUGO) method. Holub and Fridrich [5] constructed a model of individual pixel costs in Wavelet Obtained Weights (WOW) by analyzing the change in the output of directional high-pass filters produced by changing one pixel. A year later, they optimized the additive distortion on the basis of directional residuals obtained using a filter bank in their UNIversal WAvelet Relative Distortion (S-UNIWARD) tool [2]. Li *et al.* [3] proposed a cost function for the HIgh-pass, Low-pass and Low-pass (HILL) method by using a high-pass filter to locate the less predictable parts, and using two low-pass filters to make the low-cost values more clustered.

[1] https://github.com/SiaStg/SiaStegNet

Deep learning-based embedding is a growing research field. There are four major families of deep learning-based embedding [18]: (1) Via synthesis: this type of method generates images and then hides the message in them (*e.g.* Secure Steganography based on Generative Adversarial Networks, or SSGAN, by Shi *et al.* [37]), or directly considers the generated image as a stego (Hu *et al.* [38]). (2) Generating a probability map of modifications: in examples detailed by Tang *et al.* [39] (ASDL-GAN) and Yang *et al.* [40] (UT-6HPF-GAN), the generator network generates a modification map from the cover image, which makes it possible to mislead the discriminant network. (3) Via fooling powerful CNN-based steganalyzers [41], [44]: for example, Tang *et al.* [44] developed the ADV-EMB method, which adjusts the costs of modifications according to the gradients backpropagated from the target networks. (4) Via 3-player game [42], [43]: for instance, Zhu *et al.* [42] recently developed joint train encoder and decoder (HiDDeN) networks - given an input message and cover image, the encoder produces a visually indistinguishable encoded image, while the decoder can also decode the original message from it. To date, the development of deep learning-based embedding methods are still at an embryonic stage, but there are signs of potential [18].

Therefore, in this paper we propose a steganalysis method that is aimed at detecting adaptive steganography methods.

B. Existing CNN-Based Steganalysis Methods

Recent advances in deep learning and neural network techniques have made higher image steganalysis accuracy possible [28]. CNN-based steganalysis methods eliminate the need for hand-crafted features and automatically extract more extensive features from data through the use of backpropagation.

As pioneers in the field, Qian *et al.* [21] proposed a customized CNN model (which they called GNCNN), an efficient paradigm composed of three parts: preprocessing layers with non-random kernels for high-pass filtering, convolutional blocks for feature extraction, and fully-connected layers for classification. There is the first CNN-based method to achieve comparable performance with classical image steganalysis approaches using sophisticated, hand-crafted features. Xu *et al.* [23] extracted absolute values of feature map elements and truncated them using the Tanh activation function to facilitate statistical modeling. They later described several different ensemble strategies for taking advantage of a group of identically trained CNNs [24]. Yang *et al.* [25], using a process called maxCNN, demonstrated how knowledge from the selection channel can be incorporated into a CNN for steganalysis. Their process assigns large weights to features extracted from complex texture regions while assigning small weights to features learned from smooth regions. Ye *et al.* [17] introduced Spatial Rich Model (SRM) [16] filters, the knowledge of the selection channel, and developed a more appropriate activation function called the Truncated Linear Unit (TLU) in order to achieve superior performance. Yedroudj *et al.* [27] presented an improved method called Yedroudj-Net, which optimizes the structure of the neural network. By utilizing both linear and nonlinear filters, Li *et al.* [26] further enhanced the detection effect through a parallel-subnet CNN, which they named ReSTNet. Boroumand *et al.* [28] proposed SRNet, a different paradigm that randomly initializes all filters and that extracts adequate noise residual via multiple, unpooled convolutional blocks. SRNet is one of the best approaches currently available for high-detection accuracy.

Although these methods are characterized by significantly better performance compared with conventional detectors, they cannot directly train on large images due to the limitations of existing hardware. Due to the particularity of weak steganographic signals [29], resizing or cropping the images prior to classification will compromise the accuracy of the detector. To date, few methods have been developed in an attempt to solve this problem. For instance, in their proposed method, Tsang and Fridrich [29] first train a Size-Independent Detector (SID) on 256×256 image tiles; following this step, a fixed number of feature map statistical moments (*i.e.* average, minimum, maximum, and variance) substituting for final feature maps is extracted and fed into the final classifier, which consists of fully-connected layers; finally, its fully-connected layers are retrained on images of arbitrary size. Yousfi *et al.* [30] and Butora and Fridrich [31] have developed similar methodologies, but with different implementations.

In this work, we propose a different steganalysis solution for images of arbitrary size for CNNs with competitive detection accuracy. We take advantage of the relationships between image sub-regions in order to distinguish whether an image is a stego (as will be explained further in the next section).

III. Proposed Method

In Section III-A we present an overview of our method. In Section III-B we explain the motivation behind the Siamese architecture and describe related analysis. In Section III-C we present further details of our proposed network.

A. Overview

Deep learning is a relatively powerful technique. The superiority of CNN-based steganalysis techniques over all previous methods have been proven repeatedly [17], [21], [23], [25]–[28]. However, when variations are made to the content, size or lighting of the image, or to the shooting equipment, the cover images may look very different. One of the main challenges of steganalysis is to develop robust supervisory signals in order to overcome these alterations.

In this paper, we identify relationships between image sub-regions before and after the steganography operation. Fig. 2 is the flowchart of our proposed approach, which contains three phases: preprocessing, feature extraction, and fusion/classification. First, two sub-areas of the input image - 'sub_i' and 'sub_j' - separately enter the two parallel subnets. The subnets share structures, parameters, and weights. Each subnet consists of two phases: preprocessing ($\mathcal{N}$), and feature extraction ($\mathcal{F}$). At the front of each subnet, the preprocessing phase is used to produce image noise residuals ($\mathbf{n}$), which are highly related to the steganographic signal. Next, the feature

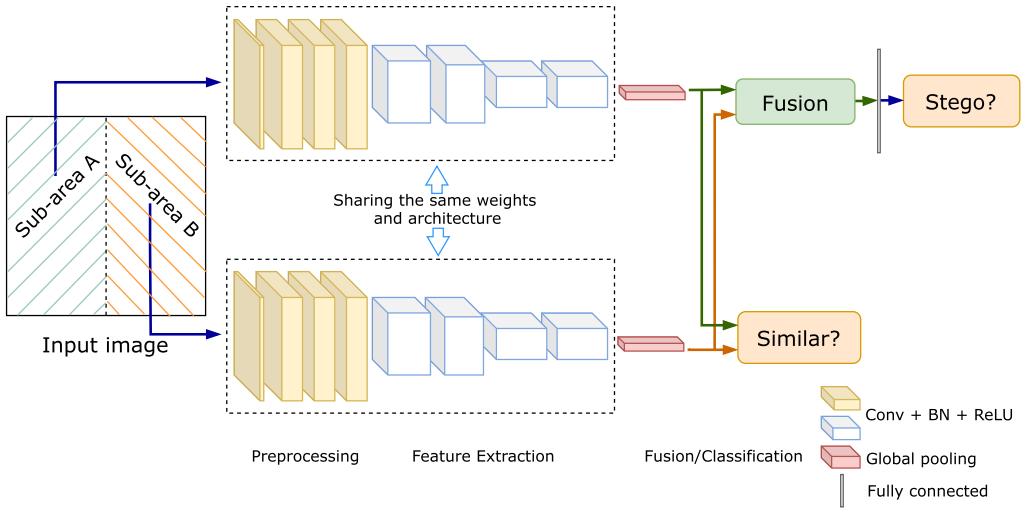


Fig. 2. Overview of our proposed network. The whole network is based on Siamese architecture. First, two sub-areas of the input image separately enter the parallel subnets. Next, the noise residuals of each sub-area are extracted. Finally, the outputs of two subnets are joined together in order to extract relationships for distinguishing stegos from covers. Detailed configurations are outlined in Fig. 5.

extraction phase is used to extract the feature vector ($\mathbf{f}$) of each sub-area noise residual. These two pieces of evidence - $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$ - are imported into a symbiotic relationship within the original image. Finally, feature vectors of the two subnets are learned under the direction of two supervisory signals in the fusion/classification phase. The first is a classification signal, which classifies the fusion of feature vectors of two subnets with probabilistic values between the stego and cover; this is achieved by means of a two-class classifier consisting of a fully-connected layer, culminating in a Softmax layer with cross-entropy loss. The second is a similarity signal, which encourages feature vectors extracted from different image sub-regions of a cover image to become similar; this is achieved via contrastive loss [46], which is based on the Euclidean distance.

The architecture of the whole network is illustrated in Section III-C. Because the whole network is data-driven, each subnet is able to describe the intrinsic properties of each image sub-area. After extracting noise features, the relationships between the outputs of two subnets help to distinguish the authenticity of the inputted image.

B. Siamese Architecture

Our network is based on Siamese architecture, which we have chosen for the following reasons:

- The adaptive steganography algorithms modify the pixel values according to the image texture, resulting in different degrees of change in different sub-areas. Their "relationship" will be changed as a result.
- Variations within cover images could overwhelm the variations between cover images and stego images, and make classification challenging. Our aim is to detect clues inside each image.

The purpose of our method is to identify the different relationships between image sub-regions in order to distinguish stegos from covers. We opine that Siamese architecture is effectively custom-tailored for this purpose since the Siamese neural network is a particular type of architecture predominantly used for tasks that involve finding similarities between two inputs. It generally consists of two symmetrical subnets, both sharing the same weights and architecture, which are joined together at the end in order to compute the similarity between the extracted features [47] and thus produce the final label. This approach was first proposed by Bromley *et al.* [48] for signature verification tasks, and is used widely for face verification tasks [49], [50].

As mentioned above, in our proposed network the steganographic noise ($\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$) features of each image sub-region (sub_i and sub_j), are extracted through each subnet of one Siamese network. The outputs of the pairwise subnets $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$ are then combined and processed as part of the final classification basis. We denote the input image as $X \in \mathbb{R}^{h \times w}$ [2], whereby each subnet $X' \in \mathbb{R}^{h' \times w'} \rightarrow \mathbf{f}_{X'} \in \mathbb{R}^n$ can be expressed in a general form:

$$\begin{aligned} \mathbf{n}_{X'} &= \mathcal{N}(X', \mathcal{W}_{\mathcal{N}}), \quad \mathbf{n}_{X'} \in \mathbb{R}^{h' \times w'} \\ \mathbf{f}_{X'} &= \mathcal{F}(\mathbf{n}_{X'}, \mathcal{W}_{\mathcal{F}}) \end{aligned} \tag{1}$$

whereby $X' \in \{sub_i, sub_j\}$, and $\mathcal{W}_{\mathcal{N}}$ and $\mathcal{W}_{\mathcal{F}}$ are sets of learnable parameters of the preprocessing phase ($\mathcal{N}$) and feature extraction phase ($\mathcal{F}$), respectively; h and w denote the height and width (respectively) of the input image, and n is the dimension of the feature vector extracted from each subnet. In our implementation, the left and right halves of the original image are sub_i and sub_j (respectively), namely $h' = h$, and $w' = w/2$.

1) Fusion Strategy: The fusion strategy is influenced by the size-independent detector (SID) described in [29]. We concatenate four non-linear moments - the maximum, minimum, mean and variance - of $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$:

- The element-wise maximum $\mathbf{m}_{max}$ of $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$;

[2]For the sake of simplicity, we omit channel and batch dimensions.

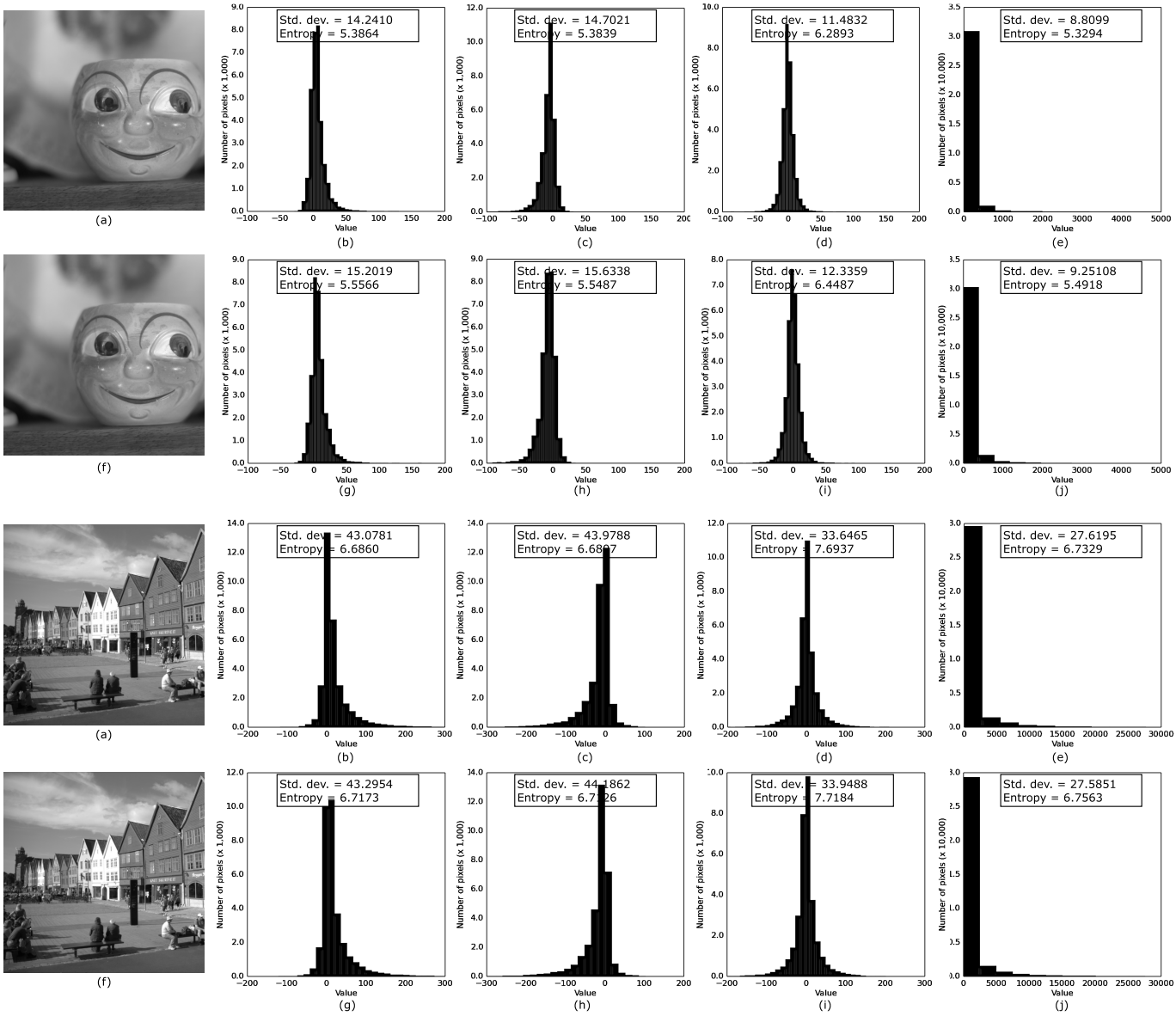


Fig. 3. Histograms of the element-wise maximum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise minimum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise mean of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, and the element-wise variance of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ before and after data embedding. The first images in the second and fourth rows are stegos. (a) The cover. (b) A histogram of the element-wise maximum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the cover. (c) A histogram of the element-wise minimum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the cover. (d) A histogram of the element-wise mean of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the cover. (e) A histogram of the element-wise variance of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the cover. (f) The corresponding stego. (g) A histogram of the element-wise maximum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the stego. (h) A histogram of the element-wise minimum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the stego. (i) A histogram of the element-wise mean of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the stego. (j) A histogram of the element-wise variance of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the stego.

- The element-wise minimum $\mathbf{m}_{min}$ of $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$;
- The element-wise mean $\mathbf{m}_{mean}$ of $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$; and,
- The element-wise variance $\mathbf{m}_{var}$ of $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$,

whereby $\mathbf{m}_{max}$, $\mathbf{m}_{min}$, $\mathbf{m}_{mean}$, $\mathbf{m}_{var} \in \mathbb{R}^n$, in order to provide information about the relationships of these two image sub-regions. This is expressed as follows:

$$Fusion(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j}) = [\mathbf{m}_{max}, \mathbf{m}_{min}, \mathbf{m}_{mean}, \mathbf{m}_{var}] \quad (2)$$

whereby $Fusion(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j}) \in \mathbb{R}^{4n}$.

For example, each dimension of variance and the range (maximum minus minimum) will be smaller for similar sub-regions, while the order moments (maximum and minimum) will be closer to the mean. Meanwhile, this fusion strategy also indirectly informs the classification phase as regards the resolution and size of the input image [29].

We embed the secret message in an example cover image shown in Fig. 3(a) using the WOW steganography method [5]. Fig. 3(f) shows the corresponding stego image. Considering that $\mathcal{N}$ is estimated using the KV kernel [21], one of the SRM kernels. Figs. 3(b) - 3(e) show histograms of the element-wise maximum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise minimum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise mean of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, and the element-wise variance of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the cover image, respectively. Figs. 3(g) - 3(j) show histograms of the

element-wise maximum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise minimum of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, the element-wise mean of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$, and the element-wise variance of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$ of the stego image, respectively. Besides, changes in entropy, as estimated using Eq. (3), reflect the increase of spatial redundancy:

$$\tilde{H} = -\sum_{i} P(\mathrm{x} = x_i) log_2 P(\mathrm{x} = x_i) \tag{3}$$

whereby

$$P(\mathrm{x} = x_i) = \frac{\text{number of elements with value } x_i}{\text{total number of elements}} \tag{4}$$

in each output of element-wise operations of $\mathbf{n}_{sub_i}$ and $\mathbf{n}_{sub_j}$. Readers should note that the learnable kernels and the other parameters of the network will continue to enhance their effectiveness in continuous neural network learning and updating; therefore, it is useful to use $Fusion(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j})$ for steganalyzing images. In our method, we extract the noise residuals for each image, and consider capturing relationships between different regions to be the primary goal of our proposed network.

2) Loss Function: The objective behind development of our network is to gain a better understanding of the connection between image sub-regions - the extent to which they are related, *i.e.* their similarity - in order to distinguish stego images. To achieve this goal, two supervised signals are designed to coincide with two aspects of image steganalysis feature construction; the label zero (0) represents covers, and label one (1) represents stegos:

- One signal is used for classification. It classifies the fused features into one of two (*i.e.* cover or stego) classification results. Classification (CLS) is achieved by (following the fusion layer, *i.e.* $Fusion(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j})$) a fully-connected layer and a 2-way softmax layer that uses cross-entropy loss:

$$L_{\mathrm{CLS}}(p, y) = \begin{cases} -log(p) & \text{if } y = 1 \\ -log(1-p) & \text{if } y = 0 \end{cases} \tag{5}$$

  whereby $y \in \{0, 1\}$ specifies the target class, and $p \in [0, 1]$ denotes the predicted probability for the class with label $y = 1$.
- The other signal is used to assess similarity. It determines whether the noise of a pair of sub-regions is the same or not. Similarity (SML) assessment is achieved by optimizing the contrastive loss [46], which is based on the Euclidean norm:

$$\begin{aligned} &L_{\mathrm{SML}}(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j}, y) \\ &= (1-y)\frac{1}{2}||\mathbf{f}_{sub_i} - \mathbf{f}_{sub_j}||_2^2 \\ &\quad + (y)\frac{1}{2}[max(0, m - ||\mathbf{f}_{sub_i} - \mathbf{f}_{sub_j}||_2)]^2 \end{aligned} \tag{6}$$

  whereby m is the margin, and $y \in \{0, 1\}$ is the binary target stating whether $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$ are similar (label 0, a cover) or not (label 1, a stego). For instance, for a cover image, this loss is low if the $\mathbf{f}_{sub_i}$ and $\mathbf{f}_{sub_j}$ are similar (closer Euclidean distance) representations, and is more substantial when they are different (farther Euclidean distance) representations.

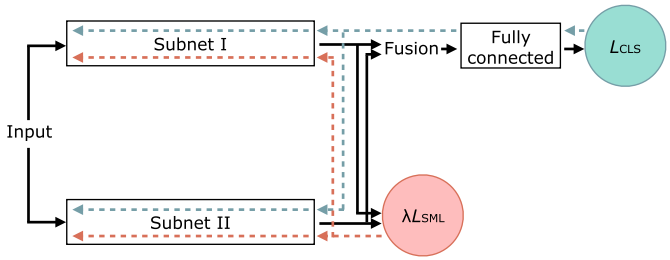


Fig. 4. Back propagation in our network. L_{CLS} affects layers of "Subnet I", "Subnet II" and "Fully connected" (green dotted arrows). L_{SML} affects layers of "Subnet I" and "Subnet II" (red dotted arrows).

These two supervised signals are weighted by hyperparameter λ in the loss function used for our network:

$$L = L_{\mathrm{CLS}}(p, y) + \lambda \cdot L_{\mathrm{SML}}(\mathbf{f}_{sub_i}, \mathbf{f}_{sub_j}, y) \tag{7}$$

Fig. 4 summarises the back propagation in our network. In Section IV-A we will investigate the interactions of L_{CLS} and L_{SML} by varying the hyperparameter λ.

It is worth emphasizing that our network is an "end-to-end" learning network, in that all of the learnable parameters (of all layers at all network phases) are trained jointly, rather than step by step. There is no extra step, such as training some layers separately.

C. Implementation Details

1) Network Architecture: As mentioned in Section III-B, stego images and cover images have different representations in terms of relationships between image sub-regions. This property can be used to distinguish whether an image is a stego.

In order to simplify implementation, as shown in Fig. 2, when an image is to be steganalyzed, we first divide each input image into two equal patches by employing a vertical line, and we take the left half and the right half as two image sub-areas in our network. The extracted relationships between them determine the final result.

The entire proposed network is organized through the Siamese architecture [48], which has been briefly introduced in Section III-B. It is worth emphasizing that it consists of two symmetrical subnets, both sharing the same weights and architecture. As mentioned, the proposed network comprises three phases: preprocessing, feature extraction, and fusion/classification.

- In the preprocessing phase, we extract the noise using the learnable SRM kernels proposed in [16], [17], [27]. To put it more precisely: each image sub-region is first passed through a convolutional (Conv) layer, whereby the weights of this layer are initialized to SRM filters (5×5 for a total of 30). Readers should note that the weights of all convolutional layers in our network are not fixed, and will be updated during the training. We denote the convolutional layers with "kernel size $\times$ kernel size"

filters and outputs ("the number of output channels" feature maps) as "Conv[kernel size]-[the number of output channels]". By way of example, this preprocessing layer is represented in Fig. 5 by the first blue box (Conv5-30). We then use some continuous, unpooled convolutional blocks in order to enhance the effect of extracting; this is introduced in SRNet [28]. Each unpooled block consists of 4 elements: (i) the 'Conv3-30's, (ii) batch normalization (BN) layers [51], (iii) rectified linear units (ReLU), and (iv) a short cut [52]. BN is adopted immediately after each Conv3-30 and before each ReLU, as depicted in the left side of Fig. 6 (Block A). We use "Conv3" because 3×3 is the smallest receptive field used to capture the notions of left/right/up/down/center, in accordance with the philosophy underpinning the VGGNet method [53]. These continuous convolutional blocks are represented by the brown boxes in Fig. 5.

- In the feature extraction phase, two residual building blocks (Fig. 6, Block A/B) are alternately applied to the features extracted during the first phase. This design has been inspired mainly by the rules of ResNet [54]. In this phase, we perform downsampling by setting a stride of 2 at the first convolutional layer of Block B. Once the feature map size is halved, we roughly double the number of filters of subsequent convolutional layers. Purple and green boxes mark the two stages (layers producing output maps of the same size (height/width/channels)) in this phase in Fig. 5. Next, a global average pooling layer enables the input image to be of any size, as it reduces the feature dimensionality from "[any feature size]-[any feature size]-[the number of channels]" to "1-1-[the number of channels]". This way, we obtain a "[the number of channels]" (*i.e.*, 128) dimensional vector as an output, which is a representation of each image sub-region.
- In the fusion/classification phase, as described in Section III-A and Section III-B, two supervisory signals are used simultaneously:
  - Contrastive loss is employed to measure the similarities between the outputs of two subnets by means of Euclidean distance.
  - Four element-wise statistical moments of two subnet outputs are calculated and concatenated. The resulting "4×[the number of channels]" (*i.e.*, $4 \times 128{=}512$)-dimensional vector, which captures information from the sub-areas and their relationships, is fed into a two-class classifier (a fully-connected layer, which culminates in a Softmax layer with cross-entropy loss). A dropout layer is also contained upstream from the classifier in order to prevent overfitting (dropout ratio set to 0.5).

Detailed configuration parameters are given in Fig. 5. The solid shortcuts are used when the input of the block and the output of the block are of the same dimensions (height/width/channels (depths)) (Fig. 6, Block A). The dotted shortcuts are used when the depth increase (Fig. 6, Block B), and the 1×1 convolutions on these projection shortcuts [54] are used to match dimensions. All convolutional layers are performed using a stride of 1, unless noted (as "/stride" when stride $\neq$ 1) otherwise. The effectiveness of our method will be discussed in detail in Section IV.

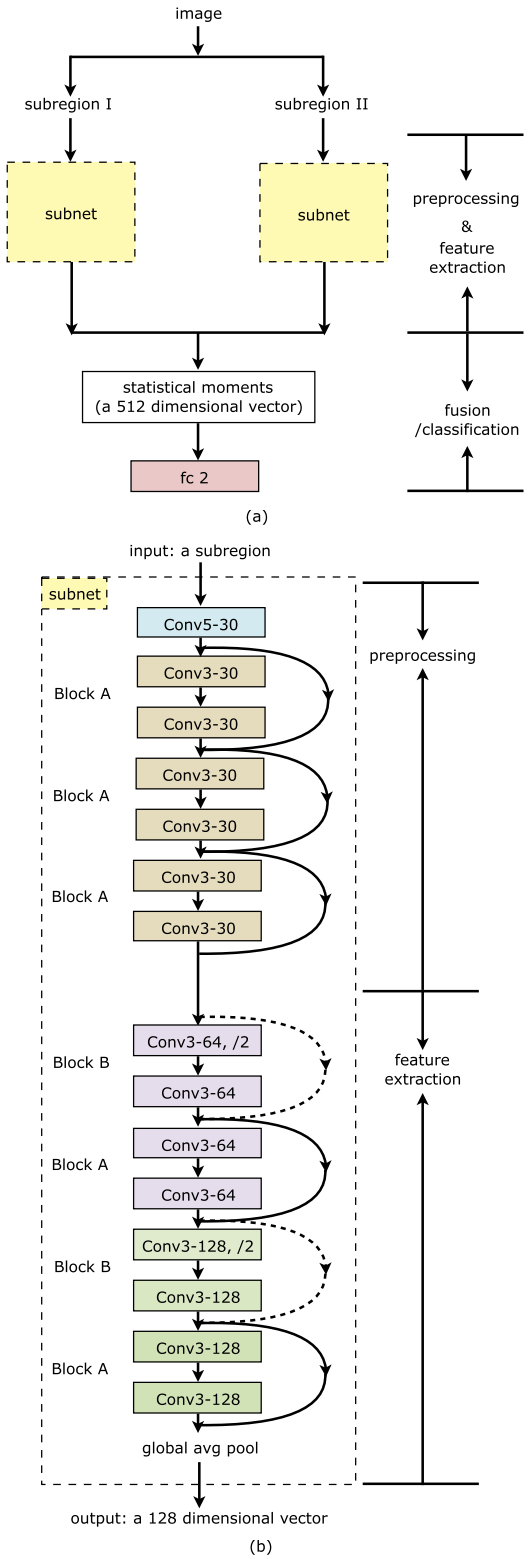


Fig. 5. (a) Architecture of our proposed network. (b) Subnet of (a). (Building blocks A and B are depicted in Fig. 6.) The convolutional layer parameters are denoted as "Conv[kernel size]-[the number of output channels] (/stride)". We represent stride as "/stride" when stride $\neq$ 1. The fully-connected layer parameters are denoted as "fc [the number of output neurons]". Some normalization layers and activation layers are omitted.

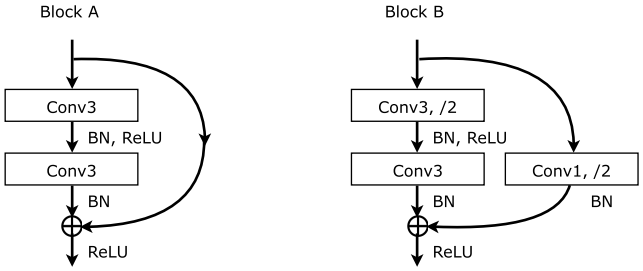
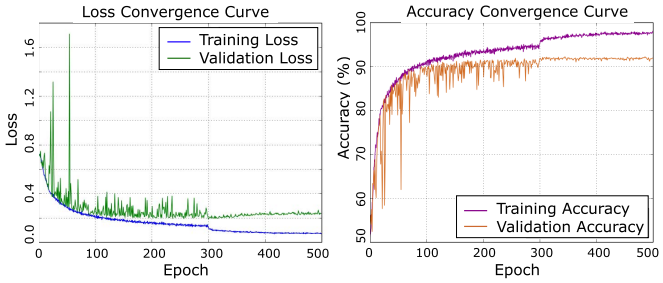


Fig. 6. Building blocks for our proposed network.



(a) (b)

Fig. 7. The loss (left) and accuracy (right) convergence plots of our proposed network when training and validating 256×256 images from the BOSSbase 1.01 produced using the S-UNIWARD method at 0.4 bpp.

2) Implementation: We use the Adamax [55] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 0.00000001$ in order to minimize contrastive loss and binary cross-entropy loss. The balancing hyperparameter λ is set to 0.1, and the margin m of contrastive loss is set to 1 (in Eq. (7)). The initial learning rate is 0.001 and is divided by 10 whenever the error plateaus. L2 regularization is used to prevent overfitting of the model. The weight decay is set to 0.0001. The network is implemented using Pytorch version 1.4.0, and the reported runtime results are obtained using Nvidia V100 GPU. The batch size is set to 32, and so requires about 6GB of GPU memory for training 256×256 images. We train the network on different datasets for 500 epochs, a process which takes about 15 hours on our configuration. All the experimental results in this paper are reported at the final iteration. For instance, when training and validating 256×256 images sourced from BOSSbase 1.01 for the S-UNIWARD method at 0.4 bpp (bits per pixel), the convergence plots of our proposed network (provided in Fig. 7) show that the loss value of all cases decreases noticeably in the first 100 epochs and reaches stability at around the 300th epoch of the validation set. The ROC curves of our proposed network, when testing 256×256 images (BOSSbase 1.01) for the WOW, S-UNIWARD, and HILL methods at 0.4 bpp, are shown in Fig. 8. It is evident that our network is computationally efficient as a 256×256 image can be judged within 0.01s.

IV. Experimental Results and Analysis

In Sub-section IV-A, we explore the impact of the hyperparameter λ (also see Sub-subsection III-B.2) and network structures on the performance of our method. In Sub-section IV-B,

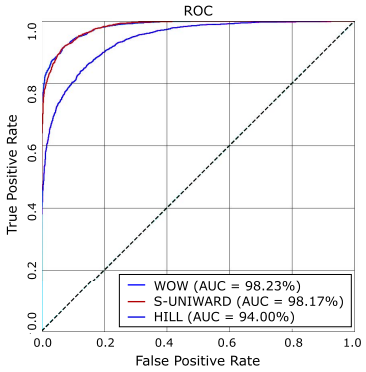


Fig. 8. ROC Curve of our proposed network when testing the 256×256 images from the BOSSbase 1.01 for the WOW, S-UNIWARD and HILL methods at 0.4 bpp.

we evaluate our method by comparing our results with those produced with SRNet [28] by considering the fixed-size benchmark dataset. In Sub-section IV-C, we present an evaluation of the arbitrary-size benchmark dataset, in which we compare it with SID [29] and an adapted SID, used as subnets of our framework. In Sub-section IV-D, we compare fixed size training and random size training. In Sub-section IV-E, we test our network using a non-content adaptive steganography method in order to verify its versatility. (For convenience, we abbreviate the name of our proposed method, Siamese Steganalysis Network, as SiaStegNet in the following paragraphs.)

A. Exploration of Hyperparameter λ and the Architecture

Training and Testing Data: We obtained cover images from 10,000 BOSSbase 1.01 native resolution images [33]. The ratios of the training set to the validation set to the testing set are 6:1:3; there is no overlap among them. They are first cropped into squares using the Irfan-View tool [56] based on the shorter side, and then resized to 256×256 using the imresize function with the bilinear interpolation algorithm in Matlab R2017a. (We abbreviate this database as BOSS_256 for convenience.) In a similar manner, we also obtain a database of size 512×512 images (named BOSS_512) using the same native resolution image training/validation/testing sets. It should be pointed out that there is thus a difference in statistical distribution between each 512×512 image and its corresponding 256×256 image. In this sub-section, our SiaStegNet is first trained on the BOSS_256 training set, and then tested on the BOSS_256 and BOSS_512 testing sets respectively without any retraining. We take these corresponding accuracy results as the output value of the objective function for tuning the network hyperparameter λ and its architecture. The testing results on the BOSS_256 set prove that SiaStegNet can steganalyze images of the same size; the results for the BOSS_512 set show its ability to transfer between images of different sizes with a slight mismatch. Fig. 9 shows some samples of the training and testing sets. For the sake of simplification, we only show the corresponding steganalysis results of S-UNIWARD [2]. In accordance with the square

TABLE I

RESULTS OF SIASTEGNET WITH DIFFERENT λ, SUGGESTING THE PRACTICABILITY OF TWO SUPERVISORY SIGNALS

Testing size	λ	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9
256×256	Accuracy(%)	91.23	91.89	91.69	91.87	91.89	91.90	91.77	91.55	91.65	91.60	91.66	91.30	91.44	91.27	90.72	90.62	90.84	90.97	91.28	90.88
	AUC(%)	97.89	98.23	98.37	98.34	98.41	98.45	98.40	98.30	98.30	98.31	98.16	98.19	98.29	98.17	97.90	98.19	98.05	98.19	98.16	98.05
512×512	Accuracy(%)	77.96	78.46	79.02	79.77	78.52	80.87	81.08	78.36	81.73	81.10	80.54	80.97	79.77	77.71	80.87	79.17	81.75	80.87	81.77	81.38
	AUC(%)	93.62	94.85	94.35	94.53	94.04	94.88	94.89	94.02	94.56	94.85	94.70	94.62	94.58	93.90	94.87	94.50	94.73	94.87	94.88	94.44

Fig. 9. Cover samples from the training and testing sets. From left, the first is a cover from the BOSS_256 training set; the second is a cover from the BOSS_256 testing set; the third is a cover from the BOSS_512 testing set. The second and third images are obtained by processing the same BOSSbase 1.01 native resolution image.

root law [57], the payload for steganography algorithms is adjusted for constant statistical detectability. The payload for 256×256 images is set to 0.4 bpp, and for 512×512 images is set to 0.24 bpp.

1) Balancing the Two Supervisory Signals: As mentioned in Section III-B, we use a Siamese network to extract relationships between image sub-regions by using the supervisory signal $L_{\text{CLS}} + \lambda \cdot L_{\text{SML}}$. In this sub-section, we consider the impact of hyperparameter λ on steganalysis performance. The search space of this parameter is set to $[0, 2)$. At $\lambda = 0$, only L_{CLS} takes effect; with increasing λ, L_{CLS} and L_{SML} dominate the training process simultaneously, and L_{SML} gradually increases its influence; at $\lambda = 1$, L_{CLS} and L_{SML} are roughly equally weighted; at λ>1, L_{SML} (is expected to) has a more significant impact on the training process.

The corresponding results with different λ values are recorded in Table I. Alongside accuracy results, and for the purpose of evaluation, Table I also contains results for the area under ROC curves (AUC). The scatter plot in Fig. 10 shows the generalization of AUC results. The dotted green and dotted yellow lines respectively indicate the AUC of detecting 256×256 and 512×512 images at $\lambda = 0$. It should be noted that the results of detecting 256×256 images are insensitive to the value of λ; in the majority of cases, using L_{SML} produces a slight gain in performance for this task, but the accuracy results produced when λ>1.3, *i.e.*, when $\lambda = 1.5$, are slightly worse. Meanwhile, L_{SML} (λ>0) favorably affects the results when the SiaStegNet trained on 256×256 images is used to steganalyze 512×512 images (whose sizes were not seen during the training). When detecting 512×512 images and when λ>0, the accuracy and AUC are significantly higher than those for when $\lambda = 0$ in most cases. It should be noted that the values of yellow points in Fig. 10 are oscillating and not increasing monotonically. In view of the above experimental evidence, and for the sake of simplicity, we set $\lambda = 0.1$ in the following experiments.

TABLE II

COMPARISONS OF SIASTEGNET AND SIASTEGNET WITH DIFFERENT ARCHITECTURES

Architecture	Testing size			
	256×256		512×512	
	Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)
SiaStegNet with a single subnet	89.44	97.56	75.80	92.39
SiaStegNet with concat layer	91.19	98.13	77.06	93.23
SiaStegNet with (double) concat layer	91.68	98.19	77.38	93.63
SiaStegNet	**91.89**	**98.23**	**78.46**	**94.85**

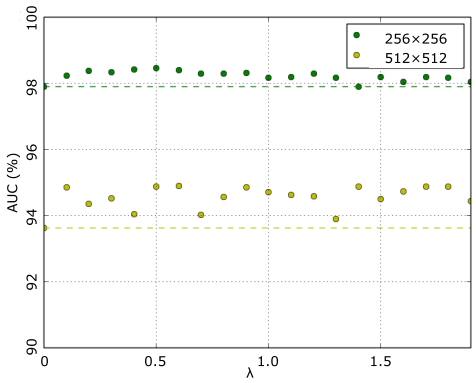


Fig. 10. The scatter plot of AUC results with different λ values, when using SiaStegNet trained on 256×256 samples to test 256×256 (the green points) and 512×512 (the yellow points) samples.

2) With Different Architectures: As shown back in Fig. 5, SiaStegNet contains two symmetrical subnets; four statistical moments of their outputs are used as the basis for classification. Here, we compare our SiaStegNet to SiaStegNets with two other different architectures: (a) SiaStegNet with a single subnet, and (b) SiaStegNet with a concat layer. Specifications of architectures (a) and (b) are outlined in Fig. 11. By comparing the results of these with the results of our SiaStegNet (in Table II), it can be seen that the backbone we architected, *i.e.*, SiaStegNet with a single subnet is successful; while the symmetrical architecture, statistical moments, and L_{SML} further improve the steganalysis performance. For example, the accuracy of detecting 256×256/512×512 images can be respectively increased by up to 1.75%/1.26% by using two symmetrical subnets, and by 2.45%/2.66% by adding statistical moments and L_{SML}. These comparisons alone demonstrate the reliability of our proposed SiaStegNet method, as well as the effectiveness of Siamese inputs (image sub-areas that respectively enter two parallel subnets) and their derived relationships.

We also compare our SiaStegNet to SiaStegNet with "double" concat layer, *i.e.*, the concat layer shown in Fig. 11 (b) is duplicated, and both operate and concatenate together.

TABLE III
COMPARISONS OF SIASTEGNET AND SRNET [28] FOR FOUR PAYLOADS IN BPP AND THREE STEGANOGRAPHY METHODS

Detector	Layers	Params	Steganography Method	0.1		0.2		0.3		0.4	
				Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)
SRNet [28]	26	4.7M	WOW	**76.97**	**86.96**	**85.93**	**94.44**	89.89	**97.29**	**92.51**	**98.72**
			S-UNIWARD	72.27	82.41	83.08	**92.91**	**88.75**	**97.01**	**92.22**	**98.67**
			HILL	69.02	76.08	**77.30**	86.56	**82.99**	**92.46**	85.57	**95.59**
SiaStegNet	**15**	**0.7M**	WOW	76.16	85.93	85.57	94.20	**89.91**	96.91	92.09	98.17
			S-UNIWARD	**72.99**	**82.42**	**83.29**	92.19	88.43	96.39	91.89	98.23
			HILL	**69.17**	**76.87**	77.26	**86.95**	82.38	91.93	**85.97**	94.00

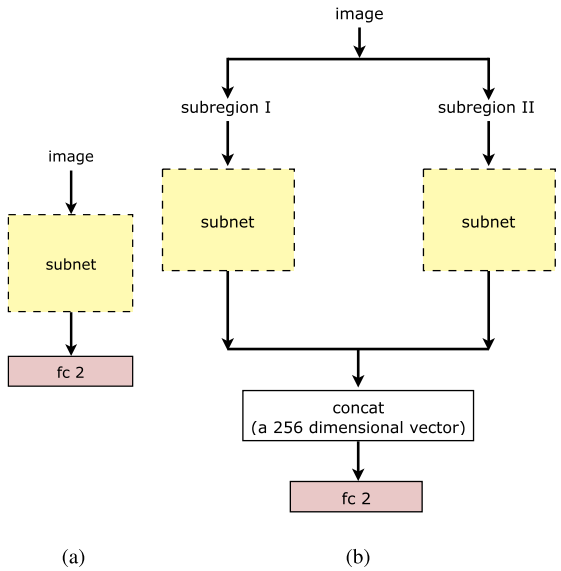


Fig. 11. Comparison of our SiaStegNet with two other different architectures: (a) SiaStegNet with a single subnet; and, (b) SiaStegNet with a concat layer. The "subnet" structure is depicted in Fig. 5 (b).

With this arrangement the number of parameters of its fully-connected layer (*i.e.*, $2 \times 256{=}512$) is the same as those of the original SiaStegNet. It should be noted that the results of accuracy and AUC of detecting 256×256 images ("SiaStegNet with (double) concat layer" in Table II) are very similar to those of the original SiaStegNet. This is mainly because all images sourced from BOSSbase 1.01 were "processed from raw files in exactly the same way (that did not include any image processing tools), all images were uncompressed, in grayscale color, with the same size" [35]. However, as shown in Table II, and consistent with the views of [29] and [30], the statistical moments are beneficial for steganalyzing multi-size databases. When detecting 512×512 images, the accuracy of SiaStegNet is 78.46%, which is 1.08% higher than of SiaStegNet with double concat layer. Therefore, we decide to use the fusion strategy using statistical moments.

Fig. 12 shows the visual feature maps on the first/last convolutional layers of the cover and corresponding stego images, and their final outputs of different subnets (which are marked by different colors). It should be noted that although the cover and stego images and their respective subregion pairs (Figs. 12 (a)/(b)) appear to be the same (through the naked eye), and the visual feature maps of the first preprocessing layer (Fig. 12 (c)) also appear similar, the visual feature maps of the last feature extraction layer (Fig. 12 (d)) are quite different. More importantly, as shown in Fig. 12 (e), there are significant differences in terms of the relationship between the final outputs of the two subnets of the cover image and the stego image. For the cover image, the output values of the two subnets are close; whereas for the corresponding stego image, they become noticeably scattered. This is achieved mainly by computing the Euclidean distance on the output of the shared subnet and optimizing the contrastive loss, while minimizing the cross-entropy loss according to the ground truth. On this basis, our SiaStegNet method is capable of performing high-quality image steganography description techniques.

B. Evaluation of the Fixed-Size Benchmark Dataset

Training and Testing Data: The experiment in this sub-section has been conducted on BOSS_256 (see Sub-section IV-A). Three adaptive steganography methods, WOW [5], S-UNIWARD [2], and HILL [3], have been employed to generate corresponding stego data, whereby the payloads are set to 0.1 bpp, 0.2 bpp, 0.3 bpp, and 0.4 bpp for each algorithm. Fig. 13 shows some cover and stego samples.

1) SiaStegNet vs. SRNet: Results for the detection accuracy of the state-of-the-art method (SRNet) [28] and our SiaStegNet method are shown in Table III [3]. It should be noted that the performance of SiaStegNet, which has fewer parameters, is comparable to that of SRNet [28]. For instance, the accuracy of our proposed SiaStegNet is 91.89% for S-UNIWARD at 0.4 bpp, which is 0.33% lower than that of the same using SRNet; and, the accuracy of SiaStegNet is 85.97% for HILL at 0.4 bpp, which is 0.40% higher than that of the same using SRNet. Both these detection accuracy percentages suggest that SiaStegNet performs well when steganalyzing fixed-size images.

C. Evaluation of the Arbitrary-Size Benchmark Dataset

Training and Testing Data: In this section, we examine differently sized images from the following databases - ALASKA_v2_TIFF_512_GrayScale, and ALASKA_v2_TIFF_VariousSize_GrayScale [35] [4] - for generating training,

[3]Somewhat surprisingly, the accuracy of SRNet regarding the BOSSbase 1.01 images in our experiments is higher than claimed in [28]. We speculate that the reason for this is that we have trained SRNet using advanced Tensorflow 1.13.0 and CUDA 10.0. Besides, the computing environment we use can perform Fused-Multiply-Add (FMA) operations [58]. The advantage of FMA is that it uses only one rounding operation instead of two, resulting in a more accurate output [59], [60].

[4] These two databases can directly be downloaded from http://alaska.utt.fr/.

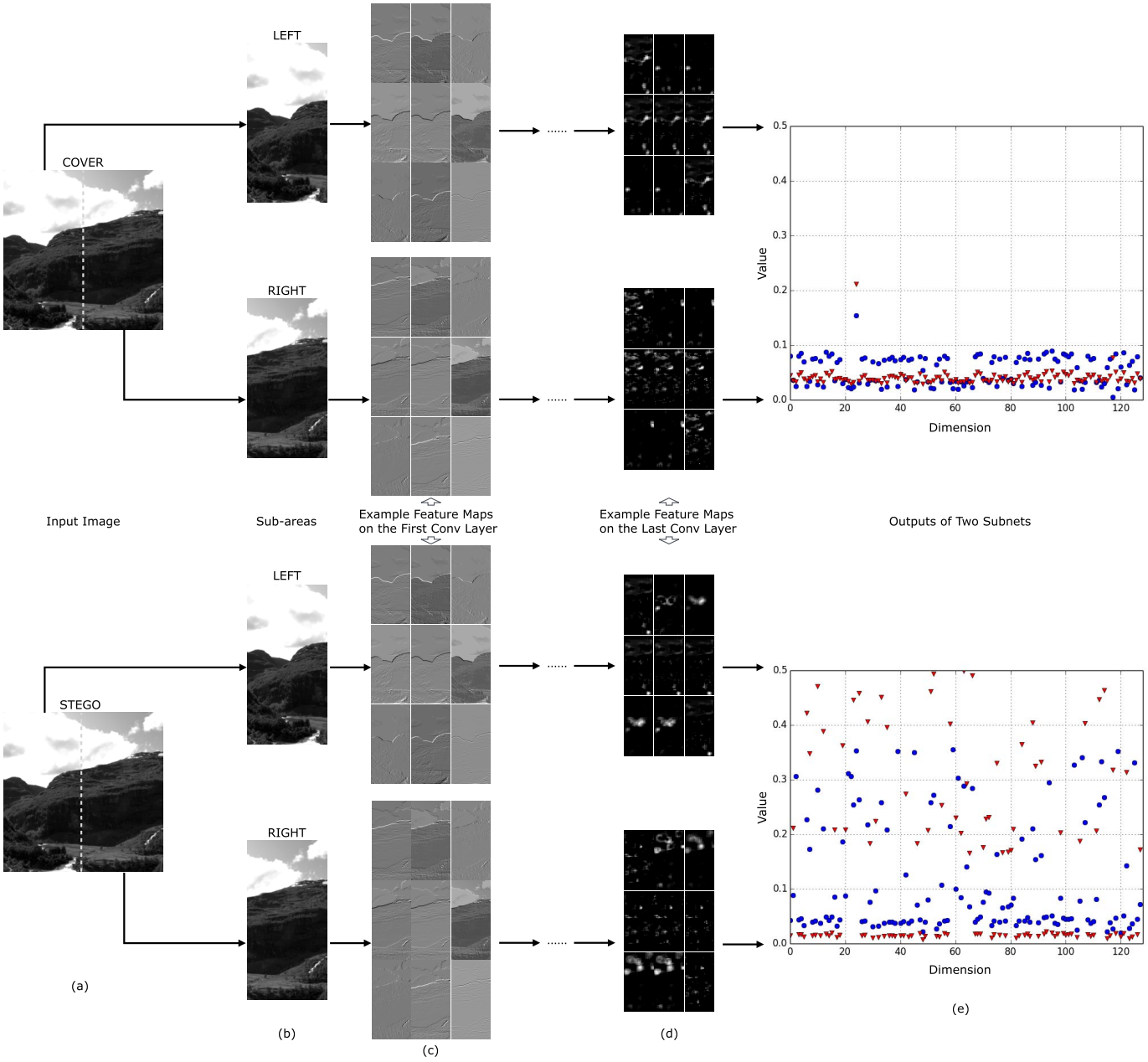


Fig. 12. Example feature maps of different subnets of a cover and the corresponding stego. The neural network learns various useful features for distinguishing stegos. With these various features being learned automatically from the proposed algorithm, our SiaStegNet method can effectively distinguish between cover and stego images. (a) Input images: the first image in the first row is a cover image, and in the second row is the corresponding stego image. (b) Sub-areas, *i.e.* the left half and the right half of input images. (c) Example feature maps on the first Conv5-30 layer. (d) Example feature maps on the last Conv3-128 layer. (e) 128-D outputs of the two subnets: the blue circle markers denote the output of the left sub-area, and the red triangle markers denote the output of the right sub-area.

validation, and testing data. We abbreviate these two databases as ALASKA_512 and ALASKA_VAR. According to the conversion script shared by the organizers of ALASKA #2 [35], ALASKA_512 is obtained by applying "smart-crop" to the images of ALASKA_VAR. "Smart-crop" is based on the selection of the area with the highest number of edges using a simple wavelet decomposition to detect edges [61]. (Fig. 14 shows five "pairs" of samples of ALASKA_512 and ALASKA_VAR.) These databases contain large repositories of images generated using many different types of camera, which have been processed in a highly heterogeneous way. ALASKA_512 contains 80005 512×512 images. ALASKA_VAR contains 16 image sizes: 512×512 (5044)[5], 512×640 (4933), 512×720 (5078), 512×1024 (4971), 640×512 (5070), 640×640 (4899), 640×720 (4927), 640×1024 (4991), 720×512 (4954), 720×640 (4972), 720×720 (5059), 720×1024 (5019), 1024×512 (5057), 1024×640 (5007), 1024×720 (5082),

[5]"Image size (total number of images of this size)"

Fig. 13. Selected samples from the training set. From left, the first is a cover image (BOSS_256); the following three stego images have been generated using adaptive steganography methods [2], [3], [5], respectively. It should be noted that differences between them are difficult to distinguish using the human eye, even though many of their pixel values have been modified.

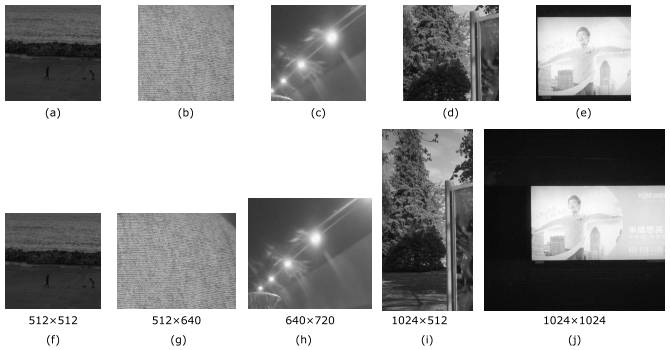


Fig. 14. Sample images of the ALASKA_512 and ALASKA_VAR databases: (a)-(e) 512×512 cover images from ALASKA_512; (f)-(i) corresponding 512×512, 512×640, 640×720, 1024×512, and 1024×1024 cover images from ALASKA_VAR.

TABLE IV

PAYLOAD (BPP) FOR S-UNIWARD/ CHANGE RATE (CHANGE PER PIXEL) FOR DIFFERENT IMAGES SIZES (HEIGHT (H) / WIDTH (W))ADJUSTED FOR CONSTANT STATISTICAL DETECTABILITY ACCORDING TO THE SQUARE ROOT LAW [57]

H \ W	512	640	720	1024
512	0.40 / 0.0633	0.35 / 0.0566	0.35 / 0.0534	0.29 / 0.0448
640	0.35 / 0.0566	0.34 / 0.0506	0.31 / 0.0477	0.28 / 0.0400
720	0.35 / 0.0534	0.31 / 0.0477	0.29 / 0.0450	0.25 / 0.0377
1024	0.29 / 0.0448	0.28 / 0.0400	0.25 / 0.0377	0.22 / 0.0317

and 1024×1024 (4942). In our approach we randomly select 60%/10% of ALASKA_512 as cover images for the training/validation sets respectively, and select (the other) 30% of ALASKA_VAR as cover images for the test set (about 1500 images of each different size). There is no overlap between training/validation/testing sets; that is, they do not contain images with the same image number. S-UNIWARD [2] is employed as a representative steganography method for generating corresponding stego data. Meanwhile, in accordance with the square root law [57], the payload for steganography algorithms is adjusted for constant statistical detectability across different sizes. Table IV summarizes the payloads and change rates used for different image sizes.

Most steganalysis approaches are designed for fixed-size images, and they therefore cannot directly train or detect patterns in large images due to the limitations of the current hardware. To solve this problem, Tsang and Fridrich [29] proposed the SID (a truly Size Independent Detector) that can steganalyze images of arbitrary size. This detector is first trained on fixed-size crops as a tile detector; then the front part of the network is fixed, and two fully-connected layers are retrained on statistical moments extracted from random crops of multiple random sizes from the training images. Our SiaStegNet offers a novel solution in that it can steganalyze images of arbitrary size whilst delivering competitive performance. It takes advantage of the relationships between image sub-regions in order to distinguish whether an image is a stego. In this sub-section, we compare our method against SID [29] [6].

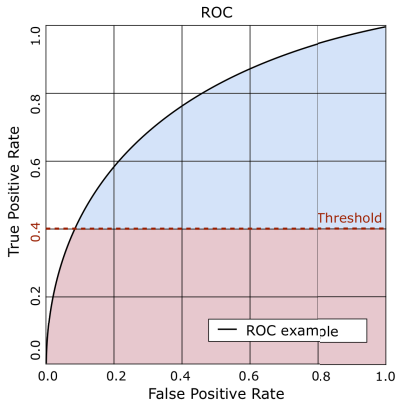


Fig. 15. The weighted AUC.

In order to give more information about the dependability of the detectors being tested on ALASKA #2 [35], and to follow its evaluation rules, results for weighted AUC (instead of AUC) are reported in this subsection in addition to accuracy. Since reliable steganalysis requires low false positive rates [15], the AUC below a true positive rate threshold is given more weight than the area above the threshold, and then normalized between 0 and 1. As illustrated in Fig. 15, the threshold is set to 0.4. The weights are 2 (below the threshold, red area) and 1 (above the threshold, blue area).

1) SiaStegNet vs. SID: Both SiaStegNet and SID are trained on images of a variety of sizes, which are prepared by random cropping being executed in the following manner. For each 512×512 image in the ALASKA_512 training set, we obtain random rectangular patches of random sizes. Both the height h and width w of the rectangle are random integers; $h, w \in [3/4 \times 512, 512]$; and the coordinates of the upper left corner of the crop (x, y), $x \in [1, 512 - w]$, and $y \in [1, 512 - h]$ is also randomly generated in order to ensure that the cropped image fits the host image. While training SID, we use hyper-parameters and training procedures according to their description, whereby it is pre-trained firstly on 512×512 image size-based set. While training SiaStegNet, we crop cover and stego samples after loading each training batch; and pre-training is not required. As mentioned above, the detectors, which are trained on small-sized images, are used directly without retraining parameters, which is in line with a more realistic situation.

[6]The backbone of SID [29] can be regarded as a modified YeNet [17].

TABLE V

RESULTS OF SID AND SIASTEGNET OF THE ARBITRARY-SIZE IMAGES. NOTE THAT DETECTORS ARE TRAINED ON 512 × 512 IMAGES. WHEN TESTING IMAGES OF OTHER SIZES, ANY NETWORK PARAMETERS WILL **NOT** BE RETRAINED

Test Image Size	Detector	Accuracy(%)	Weighted AUC(%)
512×512	SID	64.86	79.84
	SiaStegNet	**75.36**	**90.69**
512×640	SID	63.01	78.15
	SiaStegNet	**75.23**	**90.19**
640×512	SID	63.26	78.31
	SiaStegNet	**74.77**	**89.77**
512×720	SID	63.41	79.42
	SiaStegNet	**74.70**	**90.16**
720×512	SID	64.65	79.99
	SiaStegNet	**74.77**	**90.10**
640×640	SID	64.03	79.26
	SiaStegNet	**75.34**	**90.46**
640×720	SID	62.74	77.74
	SiaStegNet	**74.16**	**89.45**
720×640	SID	62.87	77.92
	SiaStegNet	**74.63**	**89.51**
720×720	SID	61.88	76.81
	SiaStegNet	**73.05**	**88.74**
512×1024	SID	61.24	76.31
	SiaStegNet	**73.10**	**88.78**
1024×512	SID	62.88	77.41
	SiaStegNet	**73.18**	**88.81**
640×1024	SID	61.21	75.65
	SiaStegNet	**73.00**	**88.39**
1024×640	SID	61.43	76.59
	SiaStegNet	**71.90**	**87.89**
720×1024	SID	60.29	75.05
	SiaStegNet	**70.53**	**86.82**
1024×720	SID	60.06	74.37
	SiaStegNet	**70.67**	**86.58**
1024×1024	SID	58.73	73.09
	SiaStegNet	**68.83**	**84.94**

Table V lists the testing results of the comparison between SiaStegNet and SID based on ALASKA_VAR. It can be seen that SiaStegNet yields better results (accuracy and weighted AUC) than SID. For example, when we steganalyze 512×512 images, the accuracy of SiaStegNet is 10.5 points higher than that of SID. When we use SiaStegNet to steganalyze images with 1.25 (512×640, 640×512) to 4 (1024×1024) times the size of the images in the training data, the accuracy is 11.08 points higher than that of SID on average. It should be noted that when the size of the images in training and testing data are different, the detection accuracy decreases, which suggests that the SiaStegNet is not totally invariant to image sizes. However, even under these conditions, SiaStegNet still performs well. When directly detecting images with 1.25 to 1.758 times (512×640 to 720×640) the size of the images in the training data, the accuracy decreases less than 2%, which demonstrates the stability of our method for steganalysis of images of arbitrary size within a limited range. This advantage helps to reduce the hardware requirements at the expense of a small amount of accuracy.

TABLE VI

RESULTS OF SID IN THE SIAMESE ARCHITECTURE OF THE ARBITRARY-SIZE IMAGES

Test Image Size	Accuracy(%)	Weighted AUC(%)
512×512	68.75	84.89
512×640	65.82	81.48
640×512	66.15	82.35
512×720	66.81	83.16
720×512	67.73	83.56
640×640	66.58	83.32
640×720	64.89	80.79
720×640	65.93	81.52
720×720	64.04	80.42
512×1024	64.03	79.39
1024×512	64.29	80.12
640×1024	63.93	79.01
1024×640	63.12	79.06
720×1024	62.33	76.24
1024×720	61.76	76.24
1024×1024	59.23	74.44

2) SID in the Siamese Architecture: Here, we use SID as the "feature predictor" (subnet) in our Siamese network in an effort to examine the network availability to different backbones. To put it more precisely: we build a network consisting of two subnets; the entire network has the same topology as SiaStegNet, and both subnets have the same structure as SID. This network is then trained (with the same two supervisory signals) and tested in the same way as SiaStegNet. Table VI lists the detection accuracy and weighted AUC results of SID in our architecture. Comparing these results with those in Table V, it can be seen that SID based in the Siamese architecture outperforms the original SID by a large margin. For example, when detecting 512×720 images, the accuracy output produced by SID in the Siamese architecture is 3.40% higher than that of the original SID, and the weighted AUC value is 3.74% higher. It may therefore be argued that our proposed framework is helpful for improving the effectiveness of the existing steganalysis network, especially on the multi-size and heterogeneous dataset.

D. Fixed Size Training vs Random Size Training

In this sub-section, we detail how much performance will SiaStegNet lose when learning on a fixed size/random sizes and testing on another image size. To put it more specifically, we demonstrate the difference between a SiaStegNet trained on random-size patches of 512×512 images, and a "dedicated" SiaStegNet trained on fixed-size samples. This experiment here has been conducted on BOSS_512 (see Sub-section IV-A). The corresponding 512×512 stego images are all embedded with the same payload of 0.24 bpp for WOW. Cover-stego pairs of other sizes are obtained by cropping from 512×512 samples without additional processing. Therefore, at this time, the statistical distribution of images of different sizes remains approximately the same.

For fixed-size training, we train our SiaStegNet to two detectors: (1) on 256×256 crops of the 512×512 images;

TABLE VII

RESULTS OF SIATEGNET WITH FIXED SIZE TRAINING VS SIATEGNET WITH RAMDOM SIZE TRAINING. ALL DETECTORS ARE TESTED ON A FIXED SIZE. PAYLOAD 0.24 BPP FOR WOW

Training	Directly tested on fixed size			
	256×256 crops of 512×512		512×512	
	Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)
256×256 crops of 512×512	83.71	92.97	89.25	96.70
512×512	82.86	91.83	**89.76**	96.82
Random crops of 512×512	**83.99**	**93.12**	89.44	**96.96**

TABLE VIII

RESULTS OF SIASTEGNET AND SID ON LSBR

Change rate	Architecture	Testing			
		BOSS_256		BOSS_512	
		Accuracy(%)	AUC(%)	Accuracy(%)	AUC(%)
0.1	SID	94.61	99.23	81.17	95.15
	SiaStegNet	**95.60**	**99.53**	**81.98**	**95.22**
0.2	SID	96.76	99.69	86.57	95.56
	SiaStegNet	**97.96**	**99.86**	**86.95**	**97.67**
0.3	SID	97.53	99.72	87.07	97.33
	SiaStegNet	**98.59**	**99.93**	**87.85**	**97.54**
0.4	SID	97.91	99.76	87.50	97.78
	SiaStegNet	**98.99**	**99.95**	**87.93**	**98.04**

and, (2) on solely 512×512 images. For random-size training, we train our SiaStegNet to one specification: (3) on random crops of the 512×512 images. (The method for obtaining random crops has been described in Section IV-C.) Then, we respectively use these three detectors to detect 256×256 crops and 512×512 images without any retraining. The results are shown in Table VII: they indicate that the accuracy of SiaStegNet when trained for random image (crop) sizes is similar to that of SiaStegNet when it is trained for a fixed image size. In addition, when the statistical distribution of testing images and training images are substantially consistent (on the rather homogeneous dataset BOSS_512), the accuracy results when their sizes are equal or unequal are almost stable.

E. Experimental Results of Non-Content Adaptive Steganography

We have also tested the detection accuracy of SiaStegNet and SID for LSBR non-content adaptive steganography [32]. The experimental results in this Sub-section are based on BOSS_256 (256×256) and BOSS_512 (512×512) (See Subsection IV-A). It should be noted there is a little mismatch in the statistical distribution (caused by resizing) between BOSS_256 images and BOSS_512 images. The change rates of BOSS_256 stegos are set to 0.1, 0.2, 0.3, 0.4 change per pixel, and the change rates of BOSS_512 stegos are scaled according to the square root law [57] to preserve the statistical detectability. First, the networks are trained on BOSS_256 samples; then, the BOSS_256 samples with the same change rate and the BOSS_512 samples with the same detectability are tested. As shown in Table VIII, the results of SiaStegNet are slightly better than those for SID. It is worth mentioning that, it is a reliable system with accuracies of over 95% and AUC values of over 99% for detecting 256×256 images, and in order to achieve such detection effects, we only trained SiaStegNet with <100 epochs with a batch size of 32.

V. CONCLUSION

In this paper, we present a Siamese CNN-based approach for steganalysis. The relationships between two image sub-regions are employed in an effort to improve steganographic feature distinction. We elaborate on the design principles of our SiaStegNet by: (i) extracting the noise features for different image sub-regions, and capturing relationships between them by using the Siamese architecture and two supervisory signals; and, (ii) systematically validating the rationality by running experiments using the BOSSbase 1.01 fixed-image size benchmark dataset. In using images sourced from the multi-sized image-containing ALASKA #2 dataset, we have demonstrated that our SiaStegNet method exhibits high transferability among different image sizes. It is also shown as a novel CNN-based framework with the potential to improve the effectiveness of existing networks for handling heterogeneous datasets.

We hope that our SiaStegNet design can provide some inspiration for future research in arbitrary-size image steganalysis, although a specific methodology of this novel framework has yet to be developed. At the same time, our future work will also focus on using the backbone of more modern networks, which will extract more valuable clues for superior distinguishing.

REFERENCES

[1] J. J. Fridrich, Ed., *Steganography in Digital Media: Principles, Algorithms, and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[2] V. Holub and J. Fridrich, "Digital image steganography using universal distortion," in *Proc. 1st ACM Workshop Inf. Hiding Multimedia Secur. IH&MMSec*, 2013, pp. 17–19.

[3] B. Li, M. Wang, J. Huang, and X. Li, "A new cost function for spatial image steganography," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2014, pp. 27–30.

[4] T. Pevný, T. Filler, and P. Bas, "Using high-dimensional image models to perform highly undetectable steganography," in *Proc. 12th Int. Conf. Inf. Hiding (IH)*, Calgary, AB, Canada, Jun. 2010, pp. 28–30.

[5] V. Holub and J. Fridrich, "Designing steganographic distortion using directional filters," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2012, pp. 234–239.

[6] V. Sedighi, R. Cogranne, and J. Fridrich, "Content-adaptive steganography by minimizing statistical detectability," *IEEE Trans. Inf. Forensics Security*, vol. 11, no. 2, pp. 221–234, Feb. 2016.

[7] S. Lyu and H. Farid, "Detecting hidden messages using higher-order statistics and support vector machines," in *Proc. 5th Int. Workshop Inf. Hiding (IH)*, Noordwijkerhout, The Netherlands, Oct. 2002, pp. 340–354.

[8] I. Avcibas, N. Memon, and B. Sankur, "Steganalysis using image quality metrics," *IEEE Trans. Image Process.*, vol. 12, no. 2, pp. 221–229, Feb. 2003.

[9] I. Avcibas, N. Memon, and B. Sankur, "Image steganalysis with binary similarity measures," in *Proc. Int. Conf. Image Process.*, 2002, pp. 645–648.

[10] J. Kodovsky, J. Fridrich, and V. Holub, "Ensemble classifiers for steganalysis of digital media," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 432–444, Apr. 2012.

[11] L. Chen, Y. Shi, P. Sutthiwan, and X. Niu, "A novel mapping scheme for steganalysis," in *Proc. 11th Int. Workshop Digit. Forensics Watermaking (IWDW)*, Shanghai, China, Oct. 2012, pp. 19–33.

[12] T. Denemark, V. Sedighi, V. Holub, R. Cogranne, and J. Fridrich, "Selection-channel-aware rich model for steganalysis of digital images," in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Dec. 2014, pp. 48–53.

[13] W. Tang, H. Li, W. Luo, and J. Huang, "Adaptive steganalysis against WOW embedding algorithm," in *Proc. 2nd ACM Workshop Inf. Hiding Multimedia Secur. IH&MMSec*, 2014, pp. 91–96.

[14] R. Cogranne, V. Sedighi, J. Fridrich, and T. Pevny, "Is ensemble classifier needed for steganalysis in high-dimensional feature spaces?" in *Proc. IEEE Int. Workshop Inf. Forensics Secur. (WIFS)*, Nov. 2015, pp. 1–6.

[15] T. Pevný and A. D. Ker, "Towards dependable steganalysis," in *Proc. Media Watermarking, Secur., Forensics*, Mar. 2015, Art. no. 94090I.

[16] J. Fridrich and J. Kodovsky, "Rich models for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 3, pp. 868–882, Jun. 2012.

[17] J. Ye, J. Ni, and Y. Yi, "Deep learning hierarchical representations for image steganalysis," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 11, pp. 2545–2557, Nov. 2017.

[18] M. Chaumont, "Deep learning in steganography and steganalysis from 2015 to 2018," in *ELSEVIER Digital Media Steganography, Principles, Algorithms, Advances*, M. Hassaballah, Ed. Amsterdam, The Netherlands: Elsevier, 2020, ch. 1, pp. 1–46.

[19] S. Tan and B. Li, "Stacked convolutional auto-encoders for steganalysis of digital images," in *Proc. Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA), Asia–Pacific*, Dec. 2014, pp. 1–4.

[20] Y. Qian, J. Dong, W. Wang, and T. Tan, "Deep learning for steganalysis via convolutional neural networks," in *Proc. Media Watermarking, Secur., Forensics*, Mar. 2015, Art. no. 94090J.

[21] Y. Qian, J. Dong, W. Wang, and T. Tan, "Learning and transferring representations for image steganalysis using convolutional neural network," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2016, pp. 25–28.

[22] L. Pibre, J. Pasquet, D. Ienco, and M. Chaumont, "Deep learning is a good steganalysis tool when embedding key is reused for different images, even if there is a cover source-mismatch," in *Proc. Media Watermarking, Secur., Forensics (MWSF)*, San Francisco, California, USA, Feb. 2016, pp. 1–11.

[23] G. Xu, H.-Z. Wu, and Y.-Q. Shi, "Structural design of convolutional neural networks for steganalysis," *IEEE Signal Process. Lett.*, vol. 23, no. 5, pp. 708–712, May 2016.

[24] G. Xu, H.-Z. Wu, and Y. Q. Shi, "Ensemble of CNNs for steganalysis: An empirical study," in *Proc. 4th ACM Workshop Inf. Hiding Multimedia Secur. IH&MMSec*, 2016, pp. 103–107.

[25] J. Yang, K. Liu, X. Kang, E. K. Wong, and Y. Shi, "Steganalysis based on awareness of selection-channel and deep learning," in *Proc. 16th Int. Workshop Digit. Forensics Watermarking (IWDW)*, Magdeburg, Germany, Aug. 2017, pp. 263–272.

[26] B. Li, W. Wei, A. Ferreira, and S. Tan, "ReST-Net: Diverse activation modules and parallel subnets-based CNN for spatial image steganalysis," *IEEE Signal Process. Lett.*, vol. 25, no. 5, pp. 650–654, May 2018.

[27] M. Yedroudj, F. Comby, and M. Chaumont, "Yedroudj-net: An efficient CNN for spatial steganalysis," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Apr. 2018, pp. 2092–2096.

[28] M. Boroumand, M. Chen, and J. Fridrich, "Deep residual network for steganalysis of digital images," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 5, pp. 1181–1193, May 2019.

[29] C. F. Tsang and J. J. Fridrich, "Steganalyzing images of arbitrary size with CNNs," *Electron. Imag.*, vol. 2018, no. 7, pp. 1–8, 2018.

[30] Y. Yousfi, J. Butora, J. Fridrich, and Q. Giboulot, "Breaking ALASKA: Color separation for steganalysis in JPEG domain," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 138–149.

[31] J. Butora and J. Fridrich, "Reverse JPEG compatibility attack," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 1444–1454, 2020.

[32] R. G. van Schyndel, A. Z. Tirkel, and C. F. Osborne, "A digital watermark," in *Proc. 1st Int. Conf. Image Process.*, Nov. 1994, pp. 86–90.

[33] P. Bas, T. Filler, and T. Pevný, "'Break our steganographic system': The ins and outs of organizing BOSS," in *Proc. 13th Int. Conf. Inf. Hiding (IH)*, Prague, Czech Republic, May 2011, pp. 59–70.

[34] R. Cogranne, Q. Giboulot, and P. Bas, "The ALASKA steganalysis challenge: A first step towards steganalysis," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 125–137.

[35] R. Cogranne, Q. Giboulot, and P. Bas. (2019). *Documentation of Alaskav2 Dataset Scripts: A Hint Moving Towards Steganography and Steganalysis into the Wild*. [Online]. Available: https://alaska.utt.fr

[36] R. Cogranne, C. Zitzmann, L. Fillatre, F. Retraint, I. Nikiforov, and P. Cornu, "A cover image model for reliable steganalysis," in *Proc. 13th Int. Conf. Inf. Hiding*, Prague, Czech Republic, May 2011, pp. 178–192.

[37] H. Shi, J. Dong, W. Wang, Y. Qian, and X. Zhang, "SSGAN: Secure steganography based on generative adversarial networks," in *Proc. Pacific–Rim Conf. Multimedia (PCM)*, Harbin, China, Sep. 2017, pp. 534–544.

[38] D. Hu, L. Wang, W. Jiang, S. Zheng, and B. Li, "A novel image steganography method via deep convolutional generative adversarial networks," *IEEE Access*, vol. 6, pp. 38303–38314, 2018.

[39] W. Tang, S. Tan, B. Li, and J. Huang, "Automatic steganographic distortion learning using a generative adversarial network," *IEEE Signal Process. Lett.*, vol. 24, no. 10, pp. 1547–1551, Oct. 2017.

[40] J. Yang, D. Ruan, J. Huang, X. Kang, and Y.-Q. Shi, "An embedding cost learning framework using GAN," *IEEE Trans. Inf. Forensics Security*, vol. 15, pp. 839–851, 2020.

[41] S. Bernard, T. Pevný, P. Bas, and J. Klein, "Exploiting adversarial embeddings for better steganography," in *Proc. ACM Workshop Inf. Hiding Multimedia Secur.*, Jul. 2019, pp. 216–221.

[42] J. Zhu, R. Kaplan, J. Johnson, and L. Fei-Fei, "Hidden: Hiding data with deep networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018, pp. 682–697.

[43] J. Hayes and G. Danezis, "Generating steganographic images via adversarial training," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, Long Beach, CA, USA, Dec. 2017, pp. 1954–1963.

[44] W. Tang, B. Li, S. Tan, M. Barni, and J. Huang, "CNN-based adversarial embedding for image steganography," *IEEE Trans. Inf. Forensics Security*, vol. 14, no. 8, pp. 2074–2087, Aug. 2019.

[45] T. Filler, J. Judas, and J. Fridrich, "Minimizing additive distortion in steganography using syndrome-trellis codes," *IEEE Trans. Inf. Forensics Security*, vol. 6, no. 3, pp. 920–935, Sep. 2011.

[46] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jun. 2006, pp. 1735–1742.

[47] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "MatchNet: Unifying feature and metric learning for patch-based matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3279–3286.

[48] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a Siamese time delay neural network," in *Proc. Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, Denver, Colorado, USA, Jun. 1993, pp. 737–744.

[49] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2005, pp. 539–546.

[50] Y. Sun, Y. Chen, X. Wang, and X. Tang, "Deep learning face representation by joint identification-verification," in *Proc. 28th Annu. Conf. Adv. Neural Inf. Process. Syst.*, Montréal, QC, Canada, Dec. 2014, pp. 1988–1996.

[51] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Jul. 2015, pp. 448–456.

[52] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," 2015, *arXiv:1505.00387*. [Online]. Available: http://arxiv.org/abs/1505.00387

[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–14.

[54] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, CA, USA, May 2015, pp. 1–15.

[56] Irfanview. *Version 4.53—64 Bit*. Accessed: May 2019. [Online]. Available: https://www.irfanview.com/

[57] A. D. Ker, T. Pevný, J. Kodovský, and J. Fridrich, "The square root law of steganographic capacity," in *Proc. 10th ACM workshop Multimedia Secur. MM&Sec*, 2008, pp. 107–116.

[58] T. Nvidia. (2017). *Nvidia Tesla V100 GPU Architecture*. [Online]. Available: http://images.nvidia.com/content/volta-architecture/pdf/volta-architecture-whitepaper.pdf

[59] N. Whitehead and A. Fit-Florea. (2011). *Precision & Performance: Floating Point and IEEE 754 Compliance for Nvidia Gpus*. [Online]. Available: https://developer.download.nvidia.com/assets/cuda/files/NVIDIA-CUDA-Floating-Point.pdf

[60] S. Markidis, S. W. D. Chien, E. Laure, I. B. Peng, and J. S. Vetter, "NVIDIA tensor core programmability, performance & precision," in *Proc. IEEE Int. Parallel Distrib. Process. Symp. Workshops (IPDPSW)*, May 2018, pp. 522–531.

[61] A. Foi, M. Trimeche, V. Katkovnik, and K. Egiazarian, "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data," *IEEE Trans. Image Process.*, vol. 17, no. 10, pp. 1737–1754, Oct. 2008.

[62] T. Filler, A. D. Ker, and J. Fridrich, "The square root law of steganographic capacity for Markov covers," in *Proc. Media Forensics Secur.*, Feb. 2009, Art. no. 725408.

[63] H. Zha, W. Zhang, C. Qin, and N. Yu, "Direct adversarial attack on stego sandwiched between black boxes," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Sep. 2019, pp. 2284–2288.

Weike You received the B.E. degree in network engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2015. She is currently pursuing the Ph.D. degree with the Institute of Information Engineering, Chinese Academy of Sciences, Beijing. Her research interests include steganography and steganalysis in digital media.

Hong Zhang (Member, IEEE) received the B.E. degree from Xiamen University in 2011 and the Ph.D. degree from the University of Chinese Academy of Sciences in 2017. He is currently an Assistant Professor with the State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences. His research interests include multimedia security, video processing, and information hiding.

Xianfeng Zhao (Senior Member, IEEE) received the Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2003. From 2003 to 2005, he was a Post-Doctoral Fellow with the Data Assurance and Communication Security Center, Chinese Academy of Sciences (CAS), Beijing. From 2006 to 2011, he was an Associate Professor with the State Key Laboratory of Information Security (SKLOIS), Institute of Software, CAS. Since 2012, he has been a Professor with SKLOIS, which was moved to the Institute of Information Engineering, CAS. Since 2013, he has also been a Professor with the University of Chinese Academy of Sciences. His research interests include information hiding and multimedia security, including steganography, steganalysis, watermarking, and multimedia forensics. In these fields, he has published more than 100 articles and owned more than 20 patents. He has served as a PC Member or the Co-Chair for many conferences, such as IWDW and AVSS. He is also an Associate Editor of the *International Journal of Digital Crime and Forensics* and an Editorial Board Member of *Forensic Science International: Reports*.